Asia Pacific Academy of Science Pte. Ltd.

Mathematics and Systems Science 2024, 2(2), 2760.
https://doi.org/10.54517/mss.v2i2.2760

Article

Research on the detection and recognition system of target vehicles based on fusion algorithm

Wenlei Wang, Chongfei Huai*, Lingyin Meng, Ziming Wang, Hao Zhang

School of Automobile and Transportation, Shenyang Ligong University, Shenyang 110159, China
*** Corresponding author:** Chongfei Huai, huaichongfei@126.com

CITATION

Wang W, Huai C, Meng L, et al. Research on the detection and recognition system of target vehicles based on fusion algorithm. Mathematics and Systems Science. 2024; 2(2): 2760. https://doi.org/10.54517/mss.v2i2.2760

ARTICLE INFO

Received: 3 June 2024
Accepted: 23 August 2024
Available online: 14 September 2024

COPYRIGHT

Copyright © 2024 by author(s). *Mathematics and Systems Science* is published by Asia Pacific Academy of Science Pte. Ltd. This work is licensed under the Creative Commons Attribution (CC BY) license. https://creativecommons.org/licenses/by/4.0/

Abstract: In modern society, the monitoring of vehicles is paramount for upholding traffic safety and order. This paper delves into a Graphical User Interface (GUI)-driven system for vehicle target detection and integration. This system seamlessly integrates three pivotal functionalities: vehicle type recognition, license plate recognition, and driver face recognition. It automates the vehicle inspection assessment process through an intuitive user interface. Specifically, for license plate recognition, the system leverages traditional computer vision techniques, including color and grayscale processing, radon transform, morphological operations, edge detection, character segmentation, and character recognition. The driver face recognition module incorporates methods such as image gray scaling, Gaussian filtering for denoising, face detection, and feature extraction. Meanwhile, vehicle type recognition is achieved by extracting Histogram of Oriented Gradients (HOG) features and utilizing a Support Vector Machine (SVM) classifier. Experimental findings highlight the system's remarkable recognition accuracy, offering a highly efficient and dependable solution for the automated inspections of vehicles.

Keywords: graphical user interface (GUI); vehicle security inspection; license plate recognition; driver face recognition; fusion algorithm

1. Introduction

Amidst the accelerating urbanization and mounting traffic pressures, intelligent traffic management systems have emerged as indispensable components in modern cities [1–4]. Vehicle type recognition, license plate recognition, and driver face recognition, as cornerstone technologies in this realm, are pivotal not only for enhancing traffic safety and efficiency, but also for vehicle management and law enforcement. Nevertheless, the market is predominated by standalone recognition systems, lacking a unified comprehensive solution, which complicates system design and diminishes data processing and integration efficiency.

Since the proposal of license plate recognition technology in the 1980s, extensive research has been conducted in this field. The primary approach has been to analyze the images of license plates to automatically extract license information and determine vehicle license numbers. Research during this period did not culminate in a complete system framework but rather focused on discussing specific issues within license plate recognition, typically resolved by employing simple image processing techniques. The recognition process involved using industrial television cameras to capture images of the front of vehicles, which were then processed by computers with simple algorithms, and still required human intervention in the final stage.

In the 1990s, with the advancement of computer vision technology and the

improvement in computer performance, systematic research on license plate recognition began to emerge. Countries such as China, the United States, Japan, and France have successively invested significant human and material resources into applied research [5]. In recent years, some countries with advanced computer and related technologies have started to explore the use of artificial neural network techniques and genetic algorithms to address the automatic recognition of license plates. Furthermore, research on the real-time requirements of license plate recognition has commenced, propelling the license plate recognition system into a practical phase. Practical License Plate Recognition (LPR) systems have also been applied in traffic monitoring, access control, and electronic toll collection, among other scenarios.

The ARGUS, developed by the image division of the British Alpha tech Company in the mid-1980s, is an automatic license plate recognition system [5] capable of processing black and white or color images. The ARGUS system takes approximately 100 milliseconds to recognize a license plate and can handle vehicles traveling at speeds of up to 100 miles per hour. Singapore's Optasia Company developed the VLPRS system [6], tailored for Singaporean license plates; Asia Vision Company in Hong Kong's license plate recognition product, VECON, is suitable for Hong Kong-style license plates. Additionally, developed countries such as Japan, Canada, Germany, Italy, and the United Kingdom have developed license plate recognition systems adapted to their national license plate formats. In terms of recognition principles, methods such as template matching, support vector machine classifiers, feature-based classifiers, artificial neural network classifiers, rough set classifiers, and cluster analysis are employed.

Research in facial recognition began in the late 1960s, with one of the early studies being the work of Bledsoe documented in reference [7], where he established a semi-automatic facial recognition system based on parameters such as the distances and ratios between facial feature points. The initial research in facial recognition primarily progressed along two directions: one was the extraction method based on facial geometric features, involving the normalization of inter-point distances and ratios of facial components, as well as the two-dimensional topological structure of the face composed of feature points like the corners of the eyes, mouth, and the tip of the nose; the other was the template matching method, which achieved recognition by calculating the self-correlation between the template and the image grayscale. In 1993, Berto provided a comprehensive introduction and comparison of these two methods and concluded that the template matching method was superior to the geometric feature method [8].Current research in facial recognition still focuses on two main directions: one is the holistic research approach, which considers the overall attributes of the pattern and includes methods such as Eigenface, SVD decomposition [9], facial iso density line analysis matching [10], elastic graph matching [11], Hidden Markov Model [12], and neural network methods; the other is the feature analysis-based method, which combines the relative ratios of facial reference points with other shape parameters or category parameters that describe facial features to form a recognition feature vector. In recent years, researchers have tended to combine holistic recognition with feature analysis, for example, the method proposed by Kin-Man Lam that

integrates analysis with the holistic approach [13], and the method proposed by Andreas Lanitis that uses flexible models to interpret and encode facial features [14].

In the field of vehicle type recognition, as early as 1986, John F. Canny introduced the Canny operator, a method for edge detection [15]. Experiments have shown that the Canny operator can retain most of the effective information while avoiding excessive redundancy. Similarly, the Sobel operator uses gradient information to detect edges and extract edge features [16], which not only yields good edge detection results but also effectively suppresses noise. John R. Smith and others proposed a method for approximating color histograms using color sets, a process that involves converting the RGB color space to a visually balanced color space, quantizing the color space, and then segmenting the image into multiple regions using color segmentation, establishing index vectors to extract image features, which can be used for preliminary image classification. In 1999, David Lowe introduced the SIFT (Scale-Invariant Feature Transform) feature extraction method, which is capable of detecting key points in an image and is invariant to scale. SIFT features are constructed based on local interest points on objects and are independent of size and rotation, as well as highly tolerant to lighting and noise. Even with limited information, the SIFT feature descriptor can extract a large number of feature vectors, and SIFT features can even be optimized to meet real-time processing requirements. The HOG (Histogram of Oriented Gradients) method focuses on extracting local texture features of an image, which involves calculating the gradient values in different directions within a specific area of the image, accumulating them to form a gradient histogram. These local features are then fed into a classifier to obtain the final classification results.

In recent years, while deep learning technology [17] has witnessed significant advancements in image recognition, traditional computer vision and image processing methods still retain their unique merits, particularly in resource-constrained environments. Current systems for vehicle type recognition, license plate recognition, and face recognition have attained maturity, exhibiting notable progress in feature extraction, image preprocessing, and classification algorithms. However, the challenge remains in effectively integrating these three technologies into a unified system while maintaining high recognition rates under varying lighting conditions, managing occlusions and complex backgrounds, and ensuring real-time processing and feedback. Thus, developing a graphical user interface (GUI) that integrates these capabilities holds significant practical value.

The novelty of this work stems from the ingenious integration of vehicle type, license plate, and driver face recognition using traditional computer vision and image processing techniques. The focus of this work is on designing a comprehensive graphical user interface (GUI) that boasts a high recognition rate. By refining feature extraction and image preprocessing algorithms, and crafting robust classification and matching strategies, an efficient, dependable, and resource-efficient recognition system [18] will be established. Furthermore, stability is a cornerstone of this design, ensuring the practicality and effectiveness of the system in real-world applications. Through this novel approach, a compelling solution for intelligent traffic management will be proposed to promote the developing of the intelligence and modernization of the traffic industry.

2. System of vehicle object detection and recognition

2.1. System workflow

This study has developed an integrated vehicle object detection and recognition system that encompasses three core functionalities: vehicle model identification, license-plate recognition, and driver face recognition. Leveraging an intuitive graphical user interface (GUI), this system automates the evaluation of vehicle inspection standards. As depicted in **Figure 1**, the system's architecture primarily comprises three stages: image input, image processing, and result output.

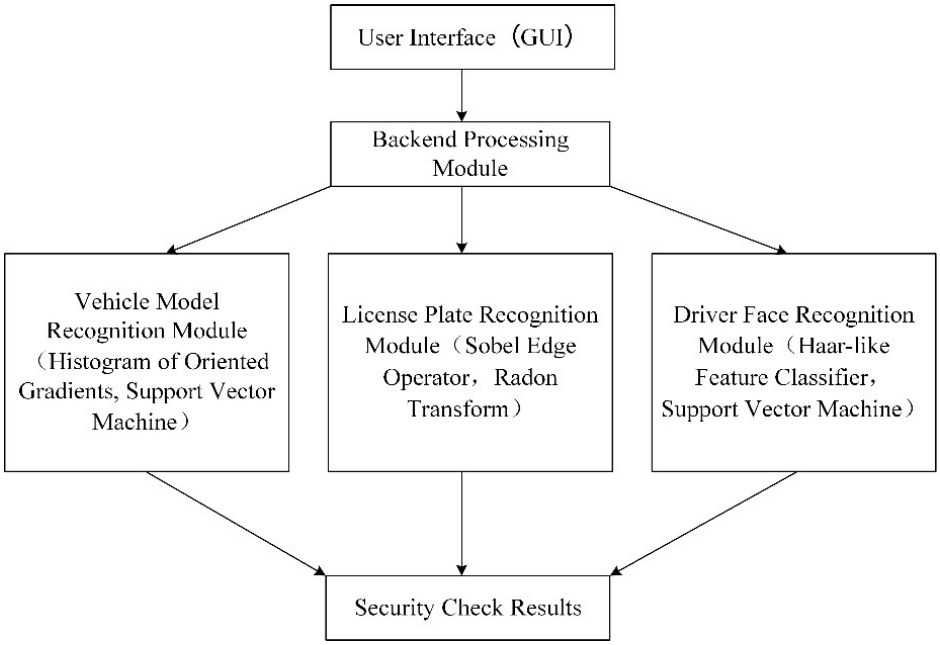

Figure 1. Diagram of system architecture.

During the image input stage, users can conveniently upload vehicle images via the GUI, which the system then receives and preliminarily processes.

The image processing stage consists of three distinct sub-modules: license-plate recognition, driver face recognition, and vehicle model identification. These modules operate sequentially, analyzing the input images and generating accurate recognition results.

Finally, in the result output stage, the system presents these recognition outcomes, including license-plate numbers, driver identity information, and the vehicle model, to the users through the intuitive GUI interface.

2.2. License-plate recognition module

The license-plate recognition module encompasses six crucial sub-steps, outlined as follows in **Figure 2**: initial coarse localization of the license plate, tilt correction to ensure proper alignment, morphological operations to enhance image clarity, precise localization of the license plate, segmentation of individual characters, and finally, character recognition to identify the license plate number.

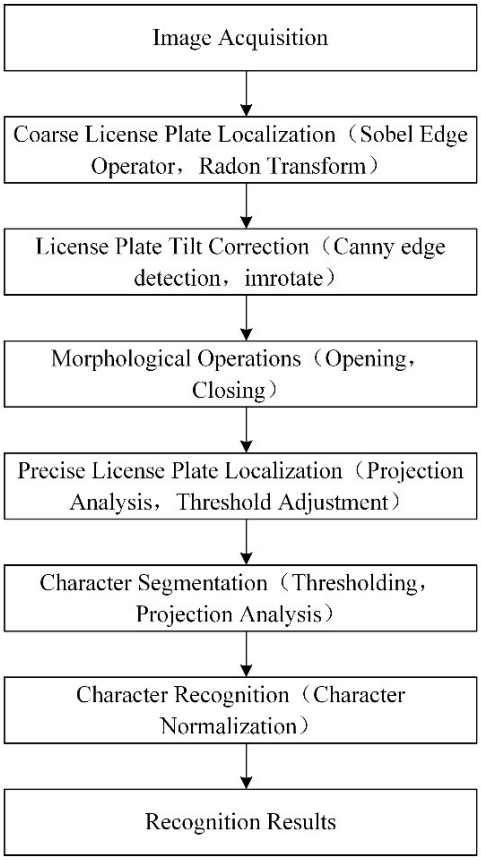

Figure 2. Workflow diagram of license-plate recognition.

2.2.1. Coarse license-plate localization

Firstly, the input color image will be converted to a grayscale image, and the principle formula [19] for the grayscale transformation is shown as below

$$Gray(i,j) = 0.299 \times R(i,j) + 0.578 \times G(i,j) + 0.114 \times B(i,j) X_1, \cdots, X_n \quad (1)$$

where *Gray* (*i, j*) represents the grayscale value of the grayscale image, while *R* (*i, j*), *G* (*i, j*) and *B* (*i, j*) represent the brightness of the red, green, and blue components, respectively. By applying the Equation (1) to the weighted average of the RGB components, a better grayscale image can be obtained, reducing interference in subsequent image processing.

By analyzing the blue pixels within the image, the approximate position of the license plate can be initially identified. The code initially counts the number of blue pixels in each row, pinpointing the row with the highest blue pixel count as the likely upper boundary of the license plate. From this starting point, the code then scans upward and downward, gradually narrowing the search until the blue pixel count drops below a predefined threshold, thus establishing the lower boundary.

To determine the lateral boundaries, the code counts the blue pixels horizontally. Initially, it estimates the positions of the left and right edges based on the typical aspect ratio of license plates. It then iteratively adjusts these estimated positions until certain specified conditions are satisfied, ensuring a precise localization.

Finally, the code crops the identified region, capturing the license plate's characteristics for subsequent refinement and recognition processes. **Figure 3**

illustrates this initial coarse localization of the license plate.

Figure 3. Coarse license plate localization.

2.2.2. License-plate tilt correction

To commence, the input color image is converted to grayscale, enabling a simpler analysis. Subsequently, the Sobel edge detection method [20] is applied to identify the distinct edges within the grayscale image. For each pixel, the algorithm calculates the weighted difference between its grayscale value and those of its immediate four neighbors (upper, lower, left, and right). This computation determines the neighborhood with the maximum weighted difference, indicating the direction of the edge closest to the pixel.

Without loss of generality, the Sobel operator can be defined as:

$$S_x = \{f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1)\} - \{f(x-1,y-1) + 2f(x-1,y) + f(x-1,y+1)\} \quad (2)$$

$$S_y = \{f(x-1,y+1) + 2f(x,y+1) + f(x+1,y+1)\} - \{f(x-1,y-1) + 2f(x,y) + f(x+1,y-1)\} \quad (3)$$

where the function $f(x, y)$ is a continuous image function. S_x represents the horizontal gradient of a pixel point, and S_y represents the vertical gradient of a pixel point.

To detect edges in an image, the Sobel operator employs two distinct kernels as depicted in **Figure 4**. One kernel is specifically designed to have the greatest sensitivity to vertical edges, while the other kernel is optimized for horizontal edges. By convolving the image with both kernels, we obtain two separate images, each highlighting a particular edge orientation. The output value for each pixel in the edge image is determined by selecting the maximum value from the convolutions of the two kernels. This approach effectively generates an image that emphasizes both horizontal and vertical edges. Subsequently, an appropriate threshold TH is selected. Any pixel $R(i, j)$ in the edge image with a value greater than or equal to TH is considered a step edge point. The Sobel operator utilizes a grayscale weighted algorithm that takes into account the pixel's upper, lower, left, and right neighborhoods. This algorithm performs edge detection based on the principle that the Sobel operator's response reaches its maximum precisely at the edge points.

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Figure 4. Sobel edge operator.

Subsequently, the code leverages the Radon transform [21] to transform the image into a sequence of projections captured at diverse angles. By comparing the projection values, the tilt angle of the image can be determined accurately.

The projection of a two-dimensional image $f(x, y)$ represents its linear summation along a particular direction. For instance, the two-dimensional line integral of $f(x, y)$ along the vertical direction corresponds to the projection of $f(x, y)$ onto the x-axis, whereas the line integral along the horizontal direction yields the projection onto the y-axis. By generalizing this concept, one can compute the projection of an image at any given angle θ, signifying that there exists a Radon transform at any angle. **Figure 5** illustrates the geometric principle underlying the Radon transformation, demonstrating how the image is projected onto lines at various angles. In a general way, the Radon transform of $f(x, y)$ is a line integral parallel to the y-axis.

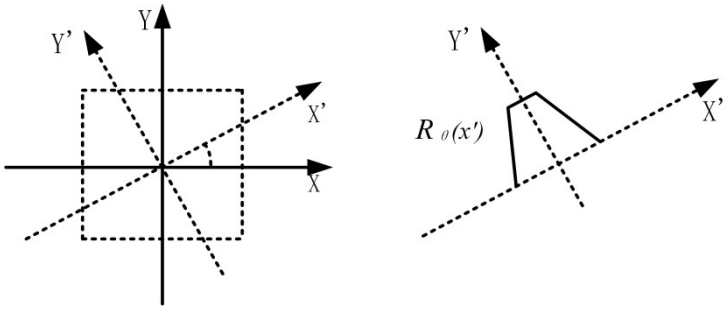

Figure 5. Schematic diagram of the geometric principle of radon transform.

$$R_\theta(x') = \int_{-\infty}^{+\infty} f(x'\cos\theta - y'\sin\theta, x'\sin\theta + y'\cos\theta)\mathrm{d}y' \tag{4}$$

where $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$. The Radon transform detects lines that correspond to peaks in the Radon transform result at point (θ, x'). The maximum value R (θ, x') is a periodic function of θ with a period T = 180°. When searching for a line, the angle θ can be limited to $0° \leq \theta \leq 180°$.

Finally, the image is rotated to align with the horizontal direction, and the subsequent blank areas resulting from the rotation are trimmed through cropping, ultimately achieving tilt correction. **Figure 6** provides a visual comparison of the images before and after the tilt correction process.

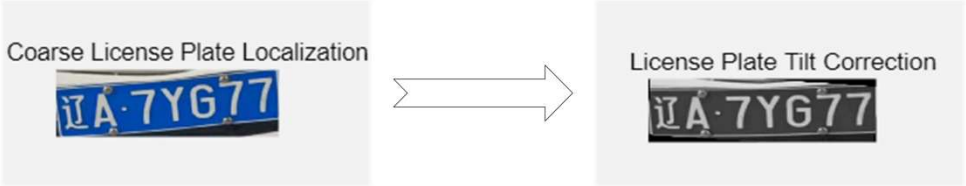

Figure 6. The image before and after tilt correction.

2.2.3. Morphological processing

After tilt correction, the input grayscale image is converted into a binary representation by applying a Dither matrix [22] for binarization. Following this, a sequence of morphological operations is executed to refine the binary image. These operations include opening, which combines erosion and dilation to remove small objects and smooth edges, as well as the removal of connected pixels and isolated pixels to further enhance the image's structural clarity.

a) Dither matrix binarization method

Employing the Ordered Dither matrix for binarization results in a dithered image. In terms of data volume, this type of image occupies only one-eighth of the space required by a pixel grayscale image, which typically utilizes 8 bits per pixel.

The Ordered Dither matrix comprises elements arranged in an arithmetic sequence, with adjacent elements having the maximum average spatial distance to ensure optimal dispersion. For instance, a 4 × 4 matrix is structured as follows:

$$T_{xy} = \begin{pmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{pmatrix} \tag{5}$$

When segmenting the image into 4 × 4 sub-blocks, each pixel within a sub-block undergoes a comparison with its corresponding matrix element in the Ordered Dither matrix. This pixel-to-element comparison determines the binarization outcome: if the pixel's grayscale value surpasses or matches the matrix element's value, the result is marked as 1; otherwise, it is labeled as 0. If the original image boasts 256 grayscale levels, the matrix element values must be scaled to accommodate this range. To evaluate the binarization's efficacy, a real-life captured image is utilized, employing the stretched matrix values from matrix T_{xy}, where

$$T_{xy} = \begin{pmatrix} 120 & 128 & 122 & 130 \\ 132 & 124 & 134 & 126 \\ 123 & 131 & 121 & 129 \\ 135 & 127 & 133 & 125 \end{pmatrix}$$

Based on the above method, the processed dithered image can be acquired and displayed in **Figure 7**.

Figure 7. Dither matrix binarization image.

2.2.4. Precise license plate localization

After the binary image undergoes morphological processing, it is converted into an inverted format, featuring a white background with black characters. Subsequently, the number of black pixels is counted horizontally and vertically to ascertain the upper and lower boundaries of the license plate. This step is followed by eliminating interference from the left border and pinpointing the precise location of the license plate [23]. Finally, the correct threshold adjustment function is employed to refine the binarization threshold, tailored specifically to the pixel area of the license plate. **Figure 8** illustrates the resulting precise localization of the license plate.

Figure 8. Precise license plate localization.

2.2.5. Character segmentation

After preprocessing, Leveraging the inherent characteristics of license plates provides valuable prior knowledge, such as: the overall length of the license plate characters spanning 409 mm, with each character having a width of 45 mm and a height of 90 mm. There is a specific 34 mm spacing between the second and third characters, while the small dot occupies a width of 10 mm and is surrounded by 12 mm gaps on either side. The remaining characters are spaced 12 mm apart. A noteworthy exception is the character "1," whose bounding rectangle differs significantly from the standard character template. While other characters' bounding rectangles adhere to the standard dimensions, the character '1' has a notably narrower width of approximately 13 mm, resulting in a spacing of about 28 mm with adjacent characters. The spacing between two consecutive '1's is approximately 44 mm. If the last character is a "1," it maintains a distance of about 28 mm from the right border. By comprehensively utilizing the aspect ratio of the license plate string, the gaps between characters, and the width-to-height ratio of each character, we can accurately segment the boundary of each individual character.

The dimensions of the license plate can be precisely determined through projection analysis. Let's define the width of the license plate as Width and its height as Text Height. Based on the prior knowledge of license plate design, the gap between the second and third characters is notably wider than the gaps between other characters. Consequently, it can considered that the left 2* width/7 portion of the license plate as the dividing line between the second and third characters. With this approach, we start by segmenting the last five characters, beginning from the third character. Once we obtain the segmentation lines for these five characters, we proceed to segment the first two characters. The single-character width of the last five characters serves as a reference standard for segmenting the Chinese characters. Remembering that the license plate typically contains seven characters, it's crucial to account for the significant amount of width occupied by the gaps between characters. As a result, it can be estimated that the prior value of the single-character width (pre-width) as 1/8 of the license plate's total width, and the prior value of the character spacing (space) as 1/32 of the license plate's width. These estimations provide a foundation for the following accurate segmentation.

Figure 9 employs character segmentation primarily relying on the predefined knowledge of character width and projection data to precisely identify the segmentation position of each character. The input for this process is a binary image that has undergone binarization and color inversion.

Figure 9. Character segmentation.

2.2.6. Character recognition

To normalize the license plate characters [24], a preprocessing step is undertaken before proceeding with character recognition. This normalization ensures that the

characters, letters, and numbers are separately identified in accordance with the sequence of the license plate number.

Character templates are categorized into distinct sets for Chinese characters, English letters, and numeric characters, each having dimensions of $M \times N$ pixels. These templates are constructed using statistical methods and stored in a database for reference.

To assess the similarity between the license plate characters and the standard templates, the author employs overlap and difference functions. Template matching involves comparing the character template with the corresponding area on the license plate to accurately recognize the character. The overlap function is defined by the Equation (6), while the difference function is given by the Equation (7). In these equations, the letters g and f represent the pixel values of the template and the corresponding license plate character area, respectively, taking values of either 0 or 1. T_f and T_g stand for the number of 1-pixels in the template and the corresponding license plate character area, respectively. C_{fg} and D_{fg} represent the overlap function and difference function, respectively. The symbols '<' and '$\oplus$' represent the AND operation and the XOR operation, respectively.

$$C_{fg} = \frac{100}{100}\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{|f_{ij} < g_{ij}|}{T_f + T_g} \tag{6}$$

$$D_{fg} = \frac{100}{100}\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{|f_{ij} \oplus g_{ij}|}{T_f + T_g} \tag{7}$$

2.3. Driver face recognition module

The driver face recognition module encompasses a comprehensive process that involves image preprocessing, precise face detection, feature extraction, and finally, face recognition, as shown in **Figure 10**.

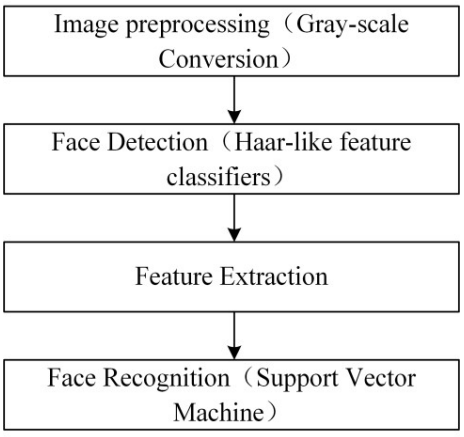

Figure 10. Schematic diagram of face recognition.

2.3.1. Image preprocessing

To improve recognition accuracy, simplify data dimensions, and reduce

computational overhead, we first convert the image to grayscale [25]. Subsequently, Gaussian filtering is applied to the grayscale image to eliminate noise. To further enhance the algorithm's robustness against non-Gaussian noise, we introduce a noise processing step based on the Minimum Error Entropy (MEE) principle after Gaussian filtering. This method, similar to the approach discussed in [Generalized Minimum Error Entropy for Robust Learning], can improve learning accuracy and robustness in the presence of various types of noise.

The Minimum Error Entropy (MEE) principle is an information theory-based method aimed at minimizing the information entropy of system output errors. Unlike traditional Mean Square Error (MSE) methods, MEE does not assume Gaussian noise distribution, thus exhibiting stronger robustness to non-Gaussian noise. The core idea of MEE is to optimize system performance by minimizing the entropy of error distribution, thereby reducing the uncertainty of system output.

In our image preprocessing, MEE is used to further reduce residual non-Gaussian noise in the image. The specific steps are as follows:

1) Calculate the difference between each pixel and its local neighborhood average as the error for the Gaussian filtered image.
2) Use kernel methods (such as Parzen window) to estimate the probability density function of these errors.
3) Calculate the entropy of errors based on the estimated probability density function.
4) Adjust filtering parameters to minimize error entropy, thus achieving optimal denoising effect.

The combination of Gaussian filtering and MEE-based noise processing enables our system to effectively handle various types of image noise. Gaussian filtering mainly processes Gaussian distributed noise, while MEE captures and reduces non-Gaussian distributed noise, providing clearer and more reliable image input for subsequent face detection and recognition processes. **Figure 11** presents a comparative visualization of the image before and after MEE processing.

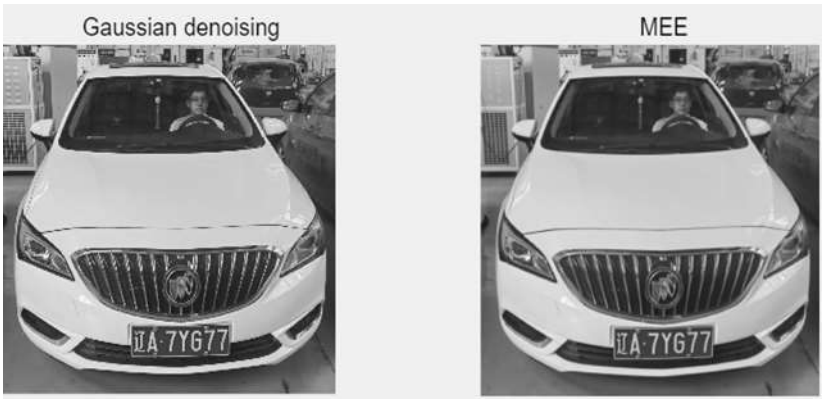

Figure 11. Comparison chart before and after MEE.

2.3.2. Face detection

Utilize Haar-like feature classifiers to identify faces in the denoised image and pinpoint the precise location of the face. Haar-like features [26] constitute a specific type of feature matrix, defined as the differential sum of pixel grayscale values

between contiguous regions within the image. Specifically, it calculates the difference between the sum of pixels in the white region and the sum of pixels in the black region, effectively capturing the gradient transition from the white to the black region. Additionally, the Haar-like rectangle feature library is further expanded, resulting in an enriched feature set, $\{f_1, f_2, \cdots, f_{15}\}$, as depicted in the **Figure 12**.

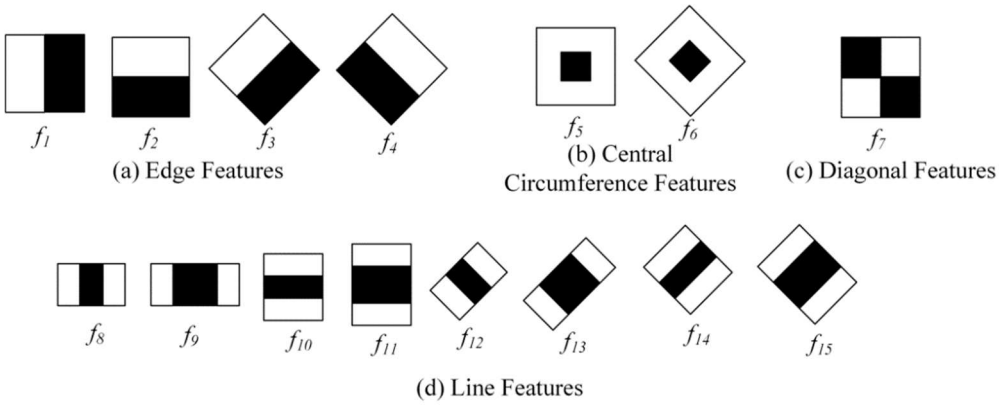

Figure 12. Haar-like matrix features.

To efficiently calculate features, an integral image is implemented. This integral image facilitates the computation of pixel sums across various regions of the image in a single pass, significantly enhancing the overall processing speed.

The definition of the integral image can be represented as follows

$$SAT(x, y) = \sum_{x' \le x, y' \le y} I(x', y') \tag{8}$$

where $I(x', y')$ represents the grayscale pixel value at position (x', y') of the image, $SAT(x, y)$ is the value of the integral image at the corresponding point.

After obtaining the integral image, taking the feature template f_1 as an example, the Harr feature values of regions A and B shown in **Figure 13** can be calculated.

Once the integral image is obtained, taking the feature template f_1 as an illustrative example, the Haar feature values of the regions A and B depicted in **Figure 13** can be readily computed.

Without loss of generality, it can be assumed that

$$R_A = SAT(i_1, j_1), R_B = SAT(i_2, j_2) - SAT(i_1, j_1) \tag{9}$$

where $SAT(i_1, j_1)$ is the sum of the pixels in rectangle A; $SAT(i_2, j_2)$ is the sum of the pixels in rectangles A and B, which are respectively the integral image values at point (i_1, j_1), (i_2, j_2). With the help of Equation (9), the feature value can be obtained as

$$R_A - R_B = 2 \times SAT(i_1, j_1) - SAT(i_2, j_2) \tag{10}$$

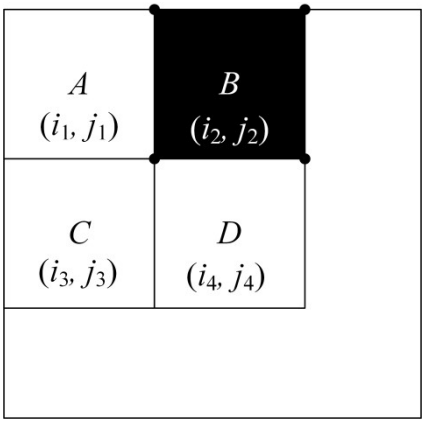

Figure 13. Example of feature value calculation.

With the introduction of the integral image, the computation of rectangular feature values becomes independent of the image's coordinates, enabling their determination solely based on the four corner coordinates of the feature template. **Figure 14** illustrates the face detection process and its outcomes.

Figure 14. Face detection diagram.

2.3.3. Face recognition

The extracted features are utilized to train a Support Vector Machine (SVM) model [27,28], which subsequently serves as the basis for recognizing face images. This trained SVM model assesses the degree of similarity between the image's features and the facial features present in the training set, ultimately predicting the categorical classification of the face within the image.

As a general learning methodology, SVM adheres to the fundamental principle of structural risk minimization, making it especially suitable for learning from small-scale samples. The essence of SVM lies in mapping input vectors to a high-dimensional feature space, where an optimal classification hyperplane is constructed. This transformation enables linearly inseparable problems in the original input space to become linearly separable in the high-dimensional space. To circumvent the 'curse of dimensionality', a kernel function $K(x_i, x_j) = \varphi(x_i)\varphi(x_j)$ is employed, serving as a surrogate for the inner product operation. Additionally, SVM leverages quadratic programming for optimization, thereby evading the pitfall of local minima.

Giving training data $(x_1, y_1), \cdots, (x1, y1), x \in R_n, y \in \{+1, -1\}$, it can be separated by a hyperplane $(\omega \cdot x) - b = 0$. When this collection of vectors is precisely partitioned by the hyperplane, without any errors, and the distance (commonly referred to as the margin) between the nearest vectors and the hyperplane is maximized, the vectors are deemed to be separated by this optimal hyperplane, also known as the maximum margin hyperplane. The decision function for this classification is:

$$f(x) = sgn\{(\omega \cdot x) - b\} \tag{11}$$

The optimal classification plane is shown in **Figure 15**, and the maximum margin is given as $\min \frac{1}{2}||w||^2$.

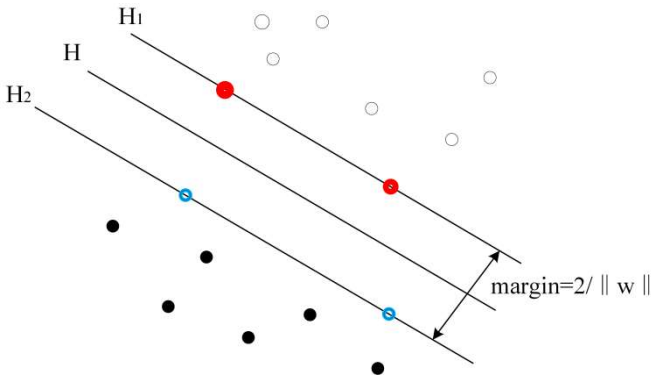

Figure 15. Optimal classification hyperplane.

Among them, all samples can be classified by

$$y_i[(w \cdot x) + b] - 1 \geq 0, i = 1,2, \cdots, n \tag{12}$$

where the points that the equality holds are the support vectors.

The discriminant function can be represented as

$$f(x) = sgn\{w^* \cdot x\} + b^*\} = sgn\{\sum_{i=1}^{n} \alpha^* y_i (x_i \cdot x) + b^*\} \tag{13}$$

For the case of non-linearly separable data, introducing the slack variable $\xi_i \geq 0$ yields:

$$y_i[(w \cdot x_i) + b] - 1 + \xi_i \geq 0, i = 1, \cdots, n \tag{14}$$

Under this constraint, it can be acquired that

$$\min \psi(w, \xi) = \frac{1}{2}(w, w) + C\left(\sum_{i=1}^{n} \xi_i\right) \tag{15}$$

where C is called the penalty parameter, which controls the punishment for misclassified samples. The support vector machine cleverly solves high-dimensional, nonlinear problems. **Figure 16** shows a diagram of the face recognition.

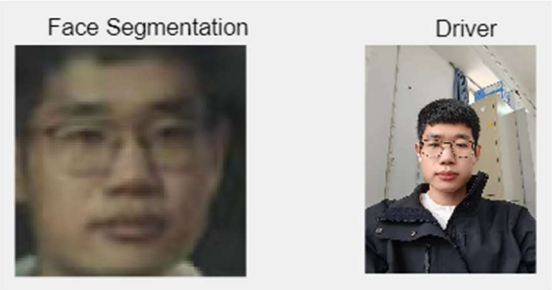

Figure 16. Face recognition diagram.

2.4. Vehicle model recognition module

The vehicle model recognition module encompasses a comprehensive process that involves image preprocessing, feature extraction, and subsequent vehicle model classification, as described in **Figure 17**.

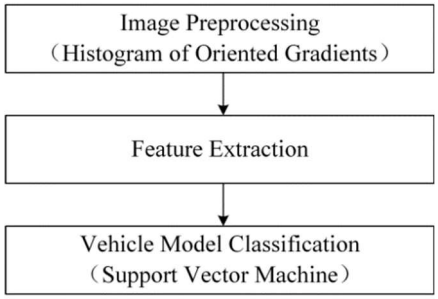

Figure 17. Vehicle model recognition schematic.

2.4.1. HOG feature extraction

For feature extraction on vehicle images, the Histogram of Oriented Gradients (HOG) method [29] is employed. HOG effectively captures the shape and contour of the image object by analyzing the distribution of edge directions. The process commences by segmenting the target image into several predefined regions of fixed size. Next, the pixel gradients within these regions are computed, and a series of calculations are performed to aggregate the gradient features. This culminates in the generation of a histogram that depicts gradient directions with a defined dimensionality, as exemplified in **Figure 18**. The entire procedure comprises the following steps.

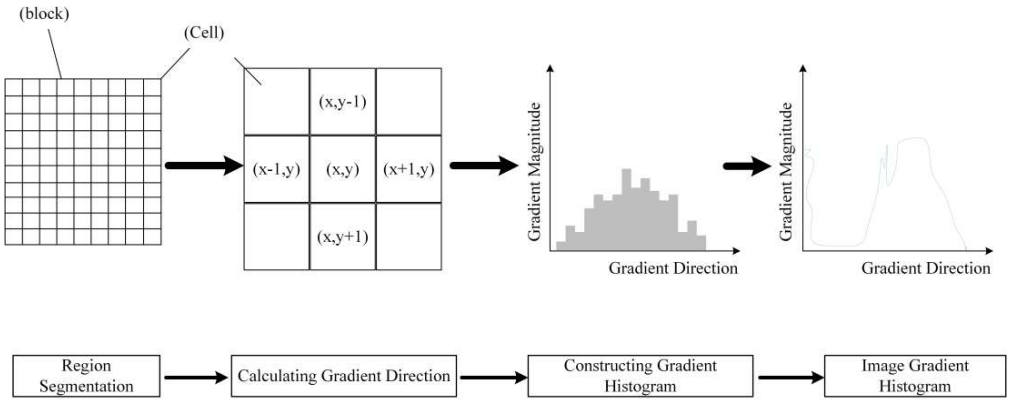

Figure 18. HOG feature extraction process.

A) Image region layer division

In the image processing phase, as outlined in reference [30], the image is meticulously divided into two distinct hierarchical layers. In the first layer, known as the initial layer, the image is subdivided into a series of interconnected units, which we refer to as Cell units. These Cell units are organized together in a specific manner, with several adjacent Cell units coming together to form a larger structural unit, which we call a Block. The design of this structure is intended to more effectively capture the local features of the image. It is particularly noteworthy that these Blocks, composed of Cell units, are not distributed in isolation within the image; there is a certain degree of overlap between them. This design feature aids in enhancing the accuracy and robustness of feature extraction, thereby capturing more detailed information from the image in subsequent processing stages.

B) Gradient value calculation

By evaluating the gradient in the coordinate direction of the pixel point (x, y), the gradient magnitude and direction of that specific point can be determined. The specific computation formulas for these are presented as

$$\begin{cases} G_x(x,y) = H(x+1,y) - H(x-1,y) \\ G_y(x,y) = H(x,y+1) - H(x,y-1) \end{cases} \tag{16}$$

In Equation (16), $G_x(x,y)$, $G_y(x,y)$, and $H(x,y)$ represent the gradients of the pixel value in the x-axis and y-axis directions in the two-dimensional plane coordinate system, respectively. The calculation formulas for the gradient magnitude and gradient direction at this pixel point can be provided as follows:

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \tag{17}$$

$$D(x,y) = arctan\left[\frac{G_y(x,y)}{G_x(x,y)}\right] \tag{18}$$

C) Construct the gradient histogram

In the context of image processing, the gradient direction for each individual Cell is meticulously categorized into n distinct bins, with each bin encompassing a range of 180 degrees. Following this categorization, the magnitudes of the gradients that fall within these specific bins are then summed up for every Cell. This accumulation process ultimately yields an n-dimensional vector that serves as a comprehensive representation of the gradient information within the Cell, capturing the essence of the gradient's orientation and strength in a multi-dimensional format.

D) Block normalization

A collection of several Cells are aggregated to construct a larger structural unit known as a Block. Once these Cells are combined into a Block, the next step involves the normalization of the contrast that exists within this newly formed Block. This normalization process ensures that the Block's contrast levels are adjusted to a standard scale, which is crucial for maintaining consistency and comparability across different Blocks within the image.

E) Collect HOG features

As depicted in **Figure 19**, the HOG (Histogram of Oriented Gradients) features

of all overlapping Blocks within the designated detection window are meticulously gathered and compiled. This comprehensive process involves the extraction of HOG features from each individual Block and the subsequent aggregation of these features into a unified dataset. This unified dataset, which encompasses the HOG features of all Blocks within the detection window, is then used to enhance the robustness and accuracy of the subsequent detection process.

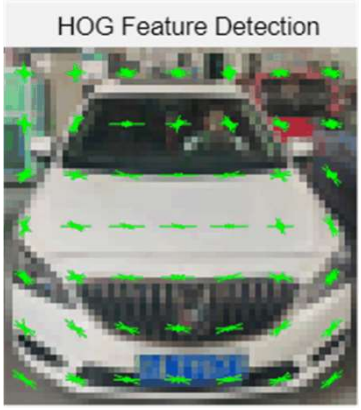

Figure 19. HOG feature extraction diagram.

2.4.2. SVM classifier

Utilize a Support Vector Machine (SVM) classifier to categorize vehicle types accurately based on the extracted HOG features. Simply input the feature vectors obtained from the HOG analysis into the trained SVM model for prediction.

2.5. System integration

This system integrates three major modules—vehicle type recognition, license plate recognition, and facial recognition—into a unified system through a Graphical User Interface (GUI). Each module employs traditional computer vision and image processing techniques to ensure efficient and stable recognition performance.

1) Modular Architecture

The system adopts a modular design, separating vehicle type recognition, license plate recognition, and facial recognition into independent submodules capable of parallel operation at different stages. Each module handles specific types of image tasks, with well-defined interfaces facilitating communication and data transfer to ensure smooth integration of the different recognition modules.

2) Data Flow Control

The image processing workflow begins with image input, followed by sequential processing through the license plate recognition module, vehicle type recognition module, and facial recognition module. Finally, the recognition results are centrally displayed in the output stage. A central controller coordinates the processing order and data flow among the modules, ensuring that each module receives the correct data at the correct time.

3) Algorithm Integration

a) License Plate Recognition:

This module combines color and grayscale processing, Radon transform,

morphological operations, edge detection, character segmentation, and character recognition techniques to ensure high accuracy in license plate recognition.

b) Facial Recognition:

The facial recognition module includes image gray scaling, Gaussian filtering for noise reduction, face detection, and feature extraction. The system can handle facial features under complex backgrounds and varying lighting conditions while maintaining stable recognition performance.

c) Vehicle Type Recognition:

The vehicle type recognition module utilizes Histogram of Oriented Gradients (HOG) feature extraction and classifies vehicle types accurately through a Support Vector Machine (SVM) classifier.

4) Recognition Results Integration

The system integrates the license plate number, vehicle type information, and driver facial recognition results, displaying them collectively in the GUI. By fusing the recognition results from each module, users can intuitively view all recognition outcomes in a unified interface.

5) Performance Optimization and Parallel Processing

The system employs parallel processing strategies, allowing multiple modules to process input data simultaneously, thereby improving processing efficiency. With scheduling algorithms and caching mechanisms, the system maintains stability and high performance even under high loads and large image inputs.

6) Error Handling and Logging System

The system integrates comprehensive error handling mechanisms, ensuring that other modules continue to operate normally even if one module encounters an error. Additionally, the system includes a logging system that records the operational status and recognition results of each module, facilitating subsequent debugging and optimization.

7) Scalability Design

The system architecture is designed with flexibility in mind, allowing for the future addition of new recognition modules or updates to existing algorithms. Users can adjust and optimize system parameters via configuration files, enabling the system to adapt to different application scenarios.

2.6. User interface design

When developing a vehicle inspection system grounded in graphical user interface [31] (GUI) technology, the focus lies in creating an intuitive and user-friendly interface that facilitates the seamless process of image input and result visualization. The pivotal design considerations for this system encompass:

1) Input Section:

Design a file selection dialog that allows users to import image or video files for processing.

Provide a preview window so that users can preview the selected files before processing.

2) Display Section:

Real-time display of processed images to ensure users can immediately see the

processing effects.

Next to the image display area, include a result text box to show the recognition results of each module in a combined text and image format.

3) Control Section:

Include buttons for start, vehicle type recognition, driver face recognition, license plate recognition, reset, and save results. These control buttons are used to manage the system's operation process.

4) Output Section:

Display the final results of the vehicle inspection on the interface, including license plate number, driver identity, and vehicle type information.

A simple and intuitive interface is provided to allow users to easily input images and view results, as shown in **Figure 20**.

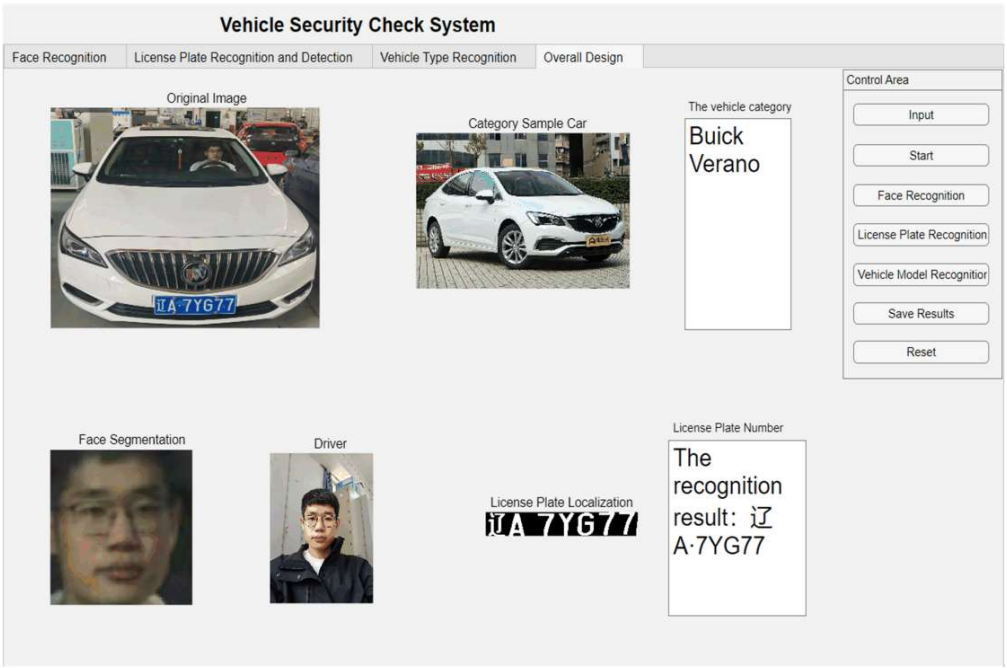

Figure 20. Vehicle security inspection system interface.

3. Experiments and results

3.1. Test platform and data sources

The experiments were conducted in a PC environment with the following configuration:

Processor: AMD Ryzen 7 5800H with Radeon Graphics (3.2 GHz)

Memory: 16GB RAM

Hard Drive: 512GB SSD

Graphics Processor: NVIDIA RTX 3050

Operating System: Microsoft Windows 11 Professional 64-bit

Development Environment: MATLAB R2023a

3.2. License plate recognition module

The experimental test images were selected from 800 captured images, with 40

license plates, each represented by 20 images. A selection of sample images from this comprehensive dataset is presented in **Figure 21**. They were divided into two major categories: good lighting conditions without shadows and poor lighting conditions with shadows, establishing two separate databases: Sample Library 1 and Sample Library 2. To further emphasize the performance of the algorithm, a comparative analysis was conducted with the Extreme Learning Machine (ELM) [32]. The experiment involved using a combination of HOG features with the widely used SVM and ELM classifiers. The test results are shown in **Table 1**. From **Table 1**, it can be observed that HOG and SVM could recognize 361 out of 400 images under good lighting conditions without shadows, while only 221 images could be recognized under shadowed conditions; HOG and ELM could recognize 344 out of 400 images under good lighting conditions without shadows, and only 201 images under shadowed conditions. The experimental results indicate that SVM has advantages in solving small sample size problems, nonlinearity, and high-dimensional pattern recognition issues.

Figure 21. Sample data images.

Table 1. License plate experimental results comparison diagram.

Model	Accuracy rate (Sample Library 1)	(Sample Library 2)
HOG + ELM	86%	50%
HOG + SVM	90%	55%

3.3. Driver face recognition results

Figure 22 represents the facial recognition module within the vehicle object detection and recognition system. To evaluate the module's performance, a comparative analysis was conducted with the Extreme Learning Machine (ELM). The experiment involved selecting 200 images from those captured and dividing them into two major categories: Sample Library 1 with good lighting conditions and no shadows, and Sample Library 2 with poor lighting conditions and shadows. In the experiment, we employed a scheme that combined HOG features with the widely used SVM and ELM classifiers. According to the test results presented in **Table 2**, we can observe

that the combination of HOG and SVM was able to recognize 65 out of 100 images under shadow-free lighting conditions, while under shadowed conditions, only 38 images were recognized. Similarly, the combination of HOG and ELM could recognize 56 images under shadow-free conditions, and only 35 images under shadowed conditions. The experimental results indicate that the combination of HOG features with SVM classifiers demonstrates superior performance in facial recognition tasks.

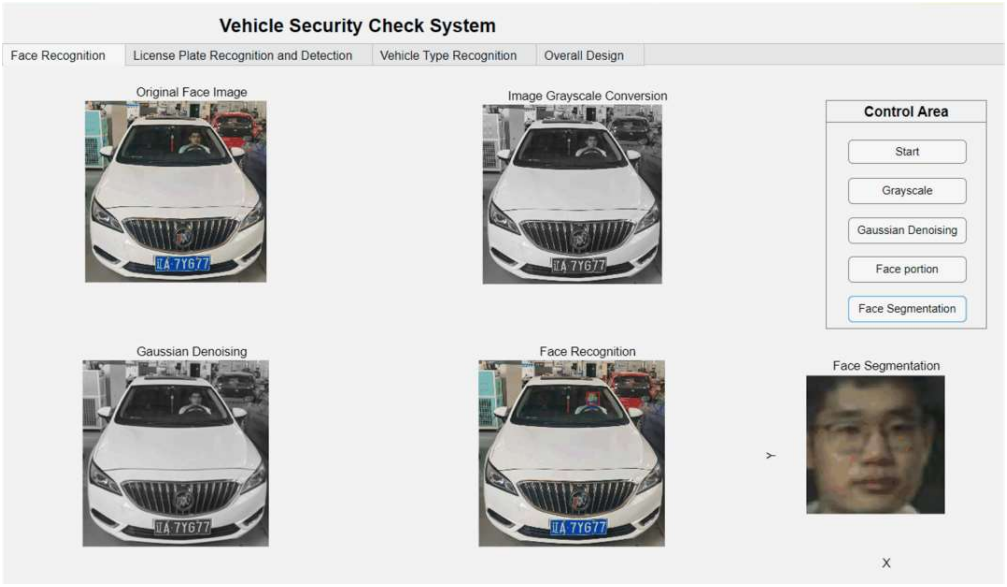

Figure 22. Facial recognition system.

Table 2. Vehicle driver face recognition results comparison diagram.

Model	Accuracy rate (Sample Library 1)	(Sample Library 2)
HOG + ELM	56%	35%
HOG + SVM	65%	38%

3.4. Vehicle model recognition results

Figure 23 depicts the vehicle model recognition module, which utilizes HOG feature extraction coupled with an SVM classifier. To train this module, a total of 147 images of cars' front views were captured and saved in *.JPG format. During testing, 90 images were assessed, with 75 of them successfully recognized, resulting in a recognition rate of 83%. Notably, for images devoid of background interference, the recognition rate surged to 94%. The experiment explored various recognition methods, including the combination of HOG features with the popular SVM [33] and ELM classifiers. As **Figure 24** illustrates, the integration of HOG features and SVM classifiers exhibited outstanding performance in the vehicle type recognition task.

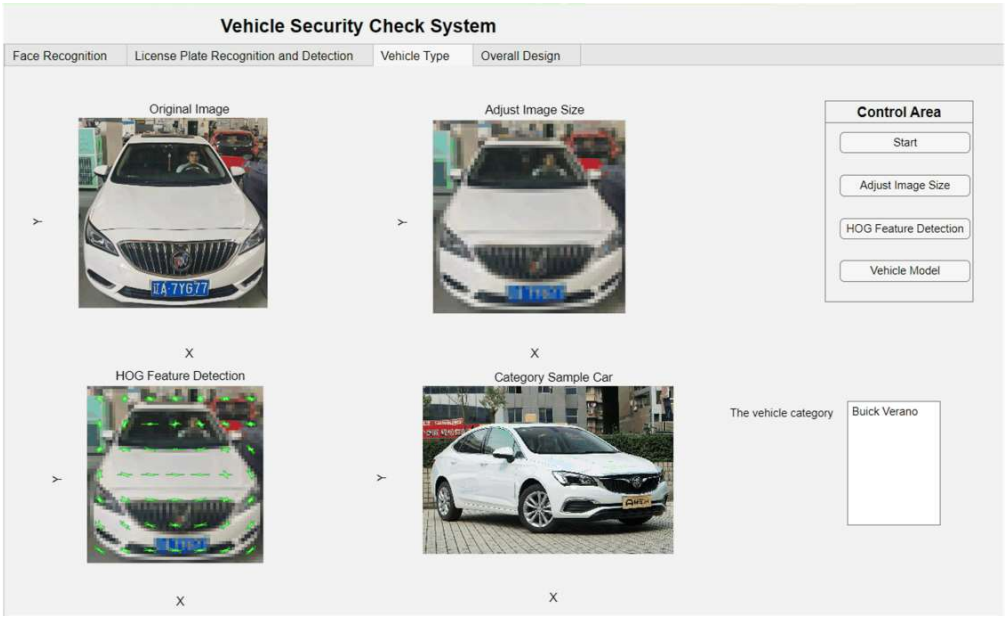

Figure 23. Vehicle model recognition system.

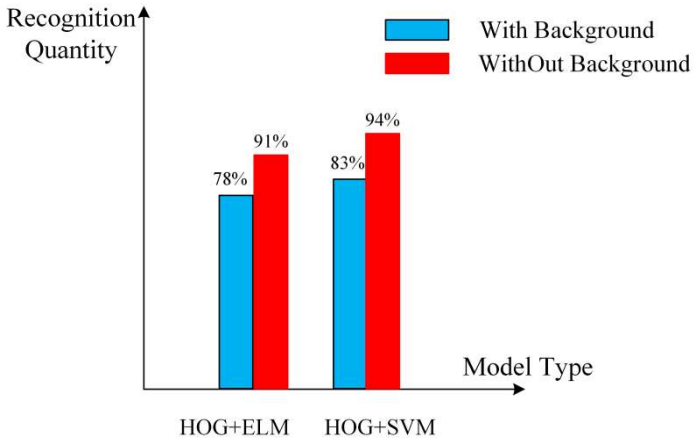

Figure 24. Vehicle model experimental results.

4. Conclusions

This study entailed the design and implementation of a comprehensive graphical user interface system for vehicle target detection, encompassing license plate recognition, vehicle type classification, and driver facial identification. In the system's modular design, the algorithms employed in each component have been thoroughly explained. To validate the system's efficacy and practical applications, a series of rigorous experimental tests have been conducted.

The experimental results yielded significant findings. Firstly, the license plate recognition accuracy attained a commendable 90%. Additionally, the facial recognition module achieved a 65% accuracy rate. In the realm of vehicle type recognition, the performance of HOG + ELM and HOG + SVM combinations were compared. The findings indicate that with HOG + ELM, the recognition rate stood at 78% with a background, surpassing to 91% in the absence of a background. In contrast, the HOG + SVM approach exhibited a recognition rate of 83% with a background, rising to an impressive 94% when the background was removed. These results

underscore the significant advantages of the HOG + SVM approach in vehicle type recognition, particularly when the background is mitigated. Through sufficient experimental validation, the vehicle object detection and recognition system developed in this work exhibited high accuracy and reliability across various target recognition tasks, suggesting its robust practicality and promising application prospects. These conclusions serve as a robust foundation for further system optimization and promotion.

Author contributions: Conceptualization, WW, CH, ZW and HZ; methodology, LM; software, WW; validation, WW; formal analysis, CH; investigation, WW; resources, LM; data curation, WW; writing—original draft preparation, WW; writing—review and editing, CH; visualization, WW; supervision, CH; project administration, WW; funding acquisition, CH. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Science and Technology Research Projects of Education Department of Liaoning Province of China (Grant No. LJKZ0264), Liaoning Province Doctoral Research Startup Fund (Grant No. 2021-BS-164).

Conflict of Interest: The authors declare no conflict of interest.

References

1. Jiang J, Li Y, Li Y, et al. Smart transportation systems using learning method for urban mobility and management in modern cities. Sustainable Cities and Society. 2024; 108105428.
2. Soumyajit G, Sourajit M, Kumar P S, et al. Two decades of vehicle make and model recognition—Survey, challenges and future directions. Journal of King Saud University Computer and Information Sciences. 2024; 36(1): 101885.
3. Tang M. Research on Image Preprocessing Algorithm of License Plate Recognition System. Advances in Computer, Signals and Systems. 2023; 7(11): 378-402.
4. Saadi I, Cunningham DW, Abdelmalik T, et al. Driver's facial expression recognition: A comprehensive survey. Expert Systems with Applications. 2024; 242122784.
5. Saha S. A Review on Automatic License Plate Recognition System. CoRR. 2019; abs/1902.09385.
6. Hou X, Fu M, Wu X, et al. Vehicle License Plate Recognition System Based on Deep Learning Deployed to PYNQ, In: Proceedings of 2018 18th International Symposium on Communications and Information Technologies (ISCIT); 26-29 September 2018; Bangkok, Thailand.
7. Bledsoe W. Man-machinefacial recognition. Panoramie Research Inc. Palo AITO, CA; 1966. p. 22.
8. Berto R, Poggio T. Face recognition: Feature versus templates. IEEE Trans. 1993; 15(10): 1042-1052.
9. Hong Z. Algebraic feature extraction of image for recognition. Pattern Recognition. 1991; 24(3): 211-219.
10. Nakamura O, Mathur S, Minami T. Identification of human faces based on isodensity maps. Pattern Recognition. 1991; 24(3): 263–272.
11. Lades M, Vorbuggen J, Buhmann J, et al. Distortion invariant object recognition in the dynamic link architecture. IEEE Trans. on Computers. 1991; 42(3): 300-311.
12. Samaria F, Young S. HMM-based architecture for face identification. Image and Vision Computing. 1994; 12(8).
13. Kin-Man L, Hong Y. An analytic-to-holistic approach for face recognition based on a single frontal view. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998; 20(7): 673-686.
14. Lanitis A, Taylor CJ, Cootes TF. Automatic interpretation and coding of face images using flexible models. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1997; 19(7): 743-756.
15. Canny JF. A theory of edge detection. IEEE Trans Pattern Anal Mach Intell. 1986; 8: 147-163.
16. Wu T, Wang L, Zhu J. Image Edge Detection Based on Sobel with Morphology. In: Proceedings of 2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC); 15-17 October 2021; Xi'an,

China.
17. Premnath SP, Gowr PS, Ananth JP, et al. Image enhancement and blur pixel identification with optimization-enabled deep learning for image restoration. Signal, Image and Video Processing. 2024; 18(5): 4525-4540.
18. Zhang X, Han B. Development of Car Security Information Management System Based on SQL Server 2000 Database Technology. Forest Engineering. 2008; (02): 92-93.
19. Chen P, Cao W, Zhang H. Research on Coarse License Plate Localization Algorithm in Complex Backgrounds. Computer Engineering and Applications. 2009; 45(17): 228-230+234.
20. Liu Z, Wang G. Research on Lane Line Recognition Based on Traditional Edge Operators (Chinese). Modern Electronics Technique. 2024; 47(07): 61-65. doi: 10.16652/j.issn.1004-373x.2024.07.010
21. Rui X. Application of Radon Transform in Skew Correction of License Plates. China Science and Technology Information. 2009; (12): 146-147.
22. José G, Paola M, Matias V, et al. A Db-Scan Binarization Algorithm Applied to Matrix Covering Problems. Computational intelligence and neuroscience. 2019; 20193238574.
23. Sharma N, Dahiya KP, Marwah RB. Performance comparison of various techniques for automatic licence-plate recognition systems. International Journal of Cloud Computing. 2022; 11(2): 138-156.
24. Sun Y, Yu H, Jia S, et al. A Character Defect Detection Algorithm Based on Edge Shape Features. Computer Applications and Software. 2023; 40(09): 177-183.
25. Pan W, Yang Y, Li H. A Digital Image Denoising Algorithm Based on Gaussian Filtering and Bilateral Filtering. ITM Web of Conferences. 2018; 1701006-01006.
26. Maale RB, Nandyal SD. Face Recognition Based on Haar Cascade Classifier. Journal of Research in Science and Engineering. 2021; 3(5): 236-254.
27. Liu Z, Wu X, Ni J, et al. Driver Intention Recognition Based on HMM and SVM Cascade Algorithm. Automotive Engineering. 2018; 40(07): 858-864.
28. Jin C, Cui Y, Wang Y. Application of Particle Swarm Optimization SVM Algorithm in Gas Analysis. Journal of Electronic Measurement and Instrumentation. 2012; 26(07): 635-639.
29. Anil J, Padma LS. A novel fast hybrid face recognition approach using convolutional Kernel extreme learning machine with HOG feature extractor. Measurement: Sensors. 2023; 30: 1076-1095.
30. Qingtian G, Haoyu Z, Fanhua Y, et al. Algorithm for Vehicle Type Recognition Based on Improved HOG Feature Extraction. Chinese Optics. 2018; 11(02): 174-181.
31. Xiao H, Dong H, Sang E, et al. Artificial Intelligence-Oriented User Interface Design and Human Behavior Recognition based on Human-Computer Nature Interaction. International Journal of Humanoid Robotics. 2023; 20(06): 203-233.
32. Chen L, Cui L, Huang R, et al. Bio-inspired neural network with application to license plate recognition: hysteretic ELM approach. Assembly Automation. 2016; 36(2): 172-178.
33. Chen S, ZhouY, Fang Y. Car Model Recognition Based on Convolutional Neural Network. Computer Applications and Software. 2017; 34(11): 228-231.